E. CIP Formats Definitions

This appendix forms an integral part of the CIP specification.

E.1 Format Definition for CIP URLs

CIP URLs are based on the RFC2056 definition for the URL structure, which defines the format for Z39.50 URLs. The Z39.50 Session and Retrieval URLs follow the Common Internet Scheme Syntax as defined in RFC 1738, "Uniform Resource Locators (URL)".

This section defines the format of a CIP URL prefix which is used in the CIP format definitions in the following sections in order to ensure that the various definitions are valid CIP URLs.

The format definition in this appendix use the following convention:

- literals are quoted with `""`;
- optional elements are enclosed in `[`brackets`]`;
- "`|`" is used to designate alternatives;
- elements may be followed with “`*`” to designate **`n`** or more repetitions of the preceding element; **`n`** defaults to **`0`**.

The low-level BNF definitions used for the format definitions are provided in Section E.10.

CIP URL Prefix Format:

```
cip_url_prefix          =     zscheme + "://" + host + [":" + port] + "/"
zscheme                 =     "z39.50s"
host                    =     u_chars_max256
                              -- The fully qualified domain name of a network
                              -- separated by “.". Fully qualified domain
                              -- names take the form as described in Section
                              -- 3.5 of RFC 1034 and Section 2.1 of RFC 1123:
                              -- a sequence of domain labels separated by
                              -- ".", each domain label starting and ending
                              -- with an alphanumeric character and
                              -- possibly also containing "-" characters. The
                              -- rightmost domain label will never start with
                              -- a digit, though, which syntactically
                              -- distinguishes all domain names from the IP
                              -- addresses. CIP B only uses domain names
port                    =     digit_max5
                              -- The port number to connect to. Most schemes
                              -- designate protocols that have a default port
                              -- number. Another port number may optionally
                              -- be supplied, in decimal, separated from the
                              -- host by a colon. If the port is omitted, the
                              -- colon is as well.
```

E.2 Database Names

There are a number of different *databases* with which a Retrieval Manager must interact. A Retrieval Manager is said to 'own' any of the *databases* that it manages and directly supports.

This section describes the format for the database names owned by a Retrieval Manager and which can be targeted for querying purposes in a Search request. The database name which is returned for each record in a Search or Present response, and which allows the exact identification of the location and the search path for the record, is presented in Section E.7.

All CIP *databases* names are formatted as Z39.50 URLs according to the following general format:

```
cip_db_name                    =    cip_url_prefix + db_name
```

The format definitions for the different kinds of *databases* are provided below:

***Explain database*:**

The *Explain database* contains the semantics of the information handled by the Retrieval Manager. The *Explain database* name is formatted as follows:

```
explain_db_name                =    cip_url_prefix + "IR-Explain-1"
```

***Extended Services database*:**

The *Extended Services database* contains the task packages created and managed by the Retrieval Manager and accessed via the use of the *Extended Services facility*. The *Extended Services database* name is formatted as follows:

```
extend_db_name                 =    cip_url_prefix + "IR-Extend-1"
```

User *database*:

The User *database* contains the user profiles created and managed by the Retrieval Manager. The User *database* is formatted as follows:

```
user_db_name                   =    cip_url_prefix + "IR-User-1"
```

CIP collections *database*:

The CIP collections *database* contains the definitions of all the collections owned (and managed) by the Retrieval Manager. The CIP collections *database* name is formatted as follows:

```
collections_db_name            =    cip_url_prefix + "IR-Collection-1"
```

Each CIP collection owned by the Retrieval Manager is considered as a *database*. A CIP collection *database* name corresponds to the name of the collection descriptor which describes the collection:

```
collection_db_name             =    url_collection_descriptor_id
```

The definition of **`collection_descriptor_id`** is provided in section E.3.

E.3 Item Descriptors Identifiers

Item descriptor identifiers are needed in order to locate or to refer to any item descriptor, i.e. collection or product descriptor, within and across Retrieval Managers. The format definitions of the item descriptors are provided hereafter:

```
url_collection_descriptor_id   =    url_reg_collection_descriptor_id |
                                    url_unreg_collection_descriptor_id

product_descriptor_id          =    product_prefix + product_id
```

```
url_reg_collection_descriptor_id   = url_archive_collection_id |
                                     url_theme_collection_id
url_unreg_collection_descriptor_id = url_persistent_result_set_id |
                                     url_persistent_query_id


url_archive_collection_id      =    cip_url_prefix + archive_collection_id
url_theme_collection_id        =    cip_url_prefix + theme_collection_id


archive_collection_id          =    archive_collection_prefix + collection_id
theme_collection_id            =    theme_collection_prefix + collection_id


archive_collection_prefix      =    "ARC_"
theme_collection_prefix        =    "THC_"
product_prefix                 =    "PID_"


collection_id                  =    u_chars_max128
product_id                     =    u_chars_max128
```

The definition of **`persistent_result_set_id`** and **`persistent_query_id`** are provided in section E.8.

E.4 Reference Identifier

Each *operation request* from an *origin* (i.e. client or Retrieval Manager) must have a unique *reference identifier*. This can be used in the tracking of an *operation* and is particularly important when a search is invoked which is passed onto remote Retrieval Managers. This means that the *request identifier* must indicate the starting *origin* of any *request* and also be unique within and beyond a single a *Z-association*, so must also indicate the date and time of the initiation of the *operation*. Finally a numeric identifier is included to identify a particular *operation*, in the situation where multiple *operations* are triggered at the same time.

It is therefore mandatory to provide the ***ReferenceId*** field in all *Z-association* objects exchanged. The format of which is an ASN.1 GeneralizedString which content is defined as:

```
reference_id                   =    a_association + z_association + timestamp +
                                    [timestamp2]


a_association                  =    origin_dns_name + a_association_id
origin_dns_name                =    u_char_max256
                                    -- DNS canonical name of the origin.
a_association_id               =    unique identifier of the a_association at the
                                    origin, e.g. counter


z_association                  =    u_chars_8
                                    -- an ASN.1 GeneralString of exactly 8
                                    -- characters which is unique within the scope
                                    -- of the Z-association.
```

timestamp	**=**	**timestamp of the initiating operation to the date/time format defined in [R10] using fine granularity (allows microsecond representation).**
timestamp2	**=**	**An optional timestamp the initial and any subsequent operation according to the date/time format defined in [R10] using fine granularity (allows microsecond representation). This to provide the capability to support performance monitoring and logging by target and origin in each operation message exchanged.**

E.5 Result Set Names

Whenever the *origin* initiates a search, a *result set name* must be defined so that the *result set* can be reused later in a *Present* or *Search service*. The *result set name* is defined by the *origin* and shall be either the *default result set name* (i.e. when a single *default result set* is maintained for a *Z-association* by the *target*) or a *named result set* (i.e. when the *target* allows the creation of several *result sets* during a *Z-association*). The *result set name* is formatted as follows:

resultset_name	**=**	**default_rs_name | named_rs_name**
default_rs_name	**=**	**"default"**
named_rs_name	**=**	**"RSN" + reference_id**

E.6 CIP APDUs

This section contains the custom CIP APDUs that are “plugged in” the appropriate EXTERNALs in the Z39.50 standard APDUs.

E.6.1 Z39.50 Extensions APDU

```
CIP-Release-B-APDU {Z39.50-CIP-B-APDU 1} DEFINITIONS ::=
BEGIN
IMPORTS IntUnit, InternationalString, Unit, ResourceReport FROM Z39.50-APDU-1995;

CIPSpecificInfo   ::=      CHOICE
   {
   searchControl             [1]       IMPLICIT  SearchControl,
   term                      [2]       IMPLICIT  CIPTerm,
   explainInfo               [3]       IMPLICIT  AdditionalSemanticAttributes
   childrenResourceReport    [4]       IMPLICIT  ChildrenResourceReport
}

SearchControl         ::=  SEQUENCE
   {
   itemDescriptorType  [1]  IMPLICIT INTEGER
                              {
                              collectionDescriptorSearch      (1),
                              productDescriptorSearch          (3)
                              }
   searchScope         [2]  IMPLICIT INTEGER
                              {
                              localSearch                     (1),
                              wideSearch                       (2)
                              }
   }
```

```
CIPTerm           ::=      CHOICE
   {
   wrsgrs              [1]  IMPLICIT    WRSGRSSpatialCoverage,
   circle              [2]  IMPLICIT     Circle,
   }

WRSGRSSpatialCoverage ::=  SEQUENCE
   {
   track               [1]  IMPLICIT    INTEGER,
   frames              [2]  SEQUENCE OF  INTEGER
   }

Circle            ::=      SEQUENCE
   {
   point               [1]  IMPLICIT    Point,
   radius              [2]  IMPLICIT    IntUnit
   }

Point             ::=      SEQUENCE
   {
   latitude            [1]  IMPLICIT    Coordinate,
   longitude           [2]  IMPLICIT    Coordinate
   }

Coordinate        ::=      InternationalString

AdditionalSemanticAttributes  ::=  SEQUENCE
   {
   shortMeaning        [1]  IMPLICIT    InternationalString        OPTIONAL,
   -- Short description of the attribute/element.
   unit                [2]  IMPLICIT    Unit                       OPTIONAL,
   -- Unit associated with the attribute/element.
   specificInstance    [3]  IMPLICIT    InternationalString        OPTIONAL,
   -- specificInstance provides textual meanings for specific values of
   -- the attribute/element.
   comment             [4]  IMPLICIT    InternationalString        OPTIONAL,
   -- comment provides additional information about the attribute/ element.
   version             [5]  IMPLICIT    InternationalString,
   -- version provides the version of the attribute/ element.
   valueSyntax         [6]  IMPLICIT    InternationalString
   -- valueSyntax is the definition of the abstract data type used for
   -- the attribute/element.
   }

ChildrenResourceReport     ::= CHOICE
   {
   terminalCollection         [1]       IMPLICIT  NULL,
   nonTerminalCollection      [2]       IMPLICIT  ChildrenReports
   }

ChildrenReports   ::=SEQUENCE OF
   {
   collectionId        [1]  IMPLICIT    InternationalString,
   collectionName      [2]  IMPLICIT    InternationalString,
   childReport         [3]  IMPLICIT    ResourceReport
   -- if the operation is a Search operation, {Z39.50-UserFormat-searchResult-1}
   -- is selected for the EXTERNAL in ResourceReport.
   -- Otherwise, {Z39.50-ResourceReport-resource-1} is selected
   }

END
```

E.6.2 CIP Ordering Extended Service APDU

This section contains the CIP Order *Extended Service APDU*, as defined in Section 3.5.8.8. It is duplicated here as a convenience for implementers.

```
{Z39.50-CIP-Order-ES} DEFINITIONS ::=
BEGIN
IMPORTS OtherInformation, InternationalString, IntUnit FROM Z39.50-APDU-1995;

CIPOrder          ::=      CHOICE
   {
   esRequest           [1]  IMPLICIT SEQUENCE{
                            toKeep     [1] OriginPartToKeep,
                            notToKeep  [2] OriginPartNotToKeep},
   taskPackage         [2]  IMPLICIT SEQUENCE{
                            originPart [1] OriginPartToKeep,
                            targetPart [2] TargetPart}
   }

OriginPartToKeep  ::=      SEQUENCE
   {
   action              [1]  IMPLICIT INTEGER {
                            orderEstimate         (1),
                            orderQuoteAndSubmit   (2),
                            orderMonitor          (3),
                            orderCancel           (4)},
   orderId             [2]  InternationalString   OPTIONAL,
   orderSpecification  [3]  OrderSpecification    OPTIONAL,
   statusUpdateOption  [4]  StatusUpdateOption    OPTIONAL,
   userInformation     [5]  UserInformation       OPTIONAL,
   otherInfo           [6]  OtherInformation      OPTIONAL
   }

OriginPartNotToKeep    ::=    SEQUENCE
   {
   orderId             [1]  InternationalString   OPTIONAL,
   orderSpecification  [2]  OrderSpecification    OPTIONAL,
   userInformation     [3]  UserInformation       OPTIONAL,
   otherInfo           [4]  OtherInformation      OPTIONAL
   }

TargetPart        ::=      SEQUENCE
   {
   orderId             [1]  InternationalString,
   orderSpecification  [2]  OrderSpecification    OPTIONAL,
   orderStatusInfo     [3]  OrderStatusInfo       OPTIONAL,
   userInformation     [4]  UserInformation       OPTIONAL,
   otherInfo           [5]  OtherInformation      OPTIONAL
   }

StatusUpdateOption     ::=    CHOICE
   {
   manual              [1]  NULL,
   automatic           [2]  IMPLICIT INTEGER {
                              eMail               (1)}
   }
```

```
UserInformation   ::=      SEQUENCE
   {
   userId              [1]  InternationalString,
   userName            [2]  InternationalString   OPTIONAL,
   userAddress         [3]  PostalAddress         OPTIONAL,
   telNumber           [4]  InternationalString   OPTIONAL,
   faxNumber           [5]  InternationalString   OPTIONAL,
   emailAddress        [6]  InternationalString   OPTIONAL,
   networkAddress      [7]  InternationalString   OPTIONAL,
   billing             [8]  Billing               OPTIONAL
   }

OrderSpecification    ::=    SEQUENCE
   {
   orderingCentreId         [1] InternationalString,
   orderPrice               [2] PriceInfo                OPTIONAL,
   orderDeliveryDate        [3] InternationalString      OPTIONAL,
   orderCancellationDate    [4] InternationalString      OPTIONAL,
   deliveryUnits            [5] SEQUENCE OF DeliveryUnitSpec,
   otherInfo                [6] OtherInformation         OPTIONAL
   }

DeliveryUnitSpec  ::=      SEQUENCE
   {
   deliveryUnitId      [1]  InternationalString   OPTIONAL,
   deliveryUnitPrice   [2]  PriceInfo             OPTIONAL,
   deliveryMethod      [3]  DeliveryMethod        OPTIONAL,
   billing             [4]  Billing               OPTIONAL,
   packages            [5]  SEQUENCE OF PackageSpec,
   otherInfo           [6]  OtherInformation      OPTIONAL
   }

DeliveryMethod    ::=      CHOICE
   {
   eMail               [1]  InternationalString,
   ftp                 [2]  FTPDelivery,
   mail                [3]  PostalAddress,
   otherInfo           [4]  OtherInformation
   }

FTPDelivery       ::=      SEQUENCE
   {
   transferDirection        [1] IMPLICIT INTEGER
                              {
                              push      (0),
                              pull      (1)
                              },
   ftpAddress          [2]  InternationalString
   }

Billing           ::=      SEQUENCE
   {
   paymentMethod       [1]  PaymentMethod,
   customerReference   [2]  IMPLICIT CustomerReference,
   customerPONumber    [3]  IMPLICIT InternationalString OPTIONAL
   }

PaymentMethod     ::=      CHOICE
   {
   billInvoice         [0]  IMPLICIT NULL,
   prepay              [1]  IMPLICIT NULL,
   depositAccount      [2]  IMPLICIT NULL,
   privateKnown        [3]  IMPLICIT NULL,
   privateNotKnown     [4]  IMPLICIT EXTERNAL},
   }
```

```
CustomerReference ::=      SEQUENCE
   {
   customerId          [1]  InternationalString,
   accounts            [2]  SEQUENCE OF InternationalString
   }

PostalAddress     ::=      SEQUENCE
   {
   streetAddress       [1]  InternationalString,
   city                [2]  InternationalString,
   state               [3]  InternationalString,
   postalCode          [4]  InternationalString,
   country             [5]  InternationalString
   }

PackageSpec       ::=      SEQUENCE
   {
   packageId           [1]  InternationalString   OPTIONAL,
   packagePrice        [2]  PriceInfo             OPTIONAL,
   package             [3]  CHOICE
                              {
                              predefinedPackage [1] PredefinedPackage,
                              adHocPackage      [2] AdHocPackage
                              },
   packageMedium       [4]  InternationalString,
   packageKByteSize    [5]  INTEGER,
   otherInfo           [6]  OtherInformation      OPTIONAL
   }

PredefinedPackage ::=      SEQUENCE
   {
   collectionId        [1]  InternationalString,
   orderItems          [2]  SEQUENCE OF OrderItem,
   otherInfo           [3]  OtherInformation      OPTIONAL
   }

AdHocPackage      ::=      SEQUENCE OF OrderItem

OrderItem         ::=      SEQUENCE
   {
   productId                [1] InternationalString,
   productPrice             [2] PriceInfo                OPTIONAL,
   productDeliveryOptions   [3] ProductDeliveryOptions   OPTIONAL,
   processingOptions        [5] ProcessingOptions        OPTIONAL,
   sceneSelectionOptions    [6] SceneSelectionOptions    OPTIONAL,
   orderStatusInfo          [7] OrderStatusInfo          OPTIONAL,
   otherInfo                [8] OtherInformation         OPTIONAL
   }

ProductDeliveryOptions::=  SEQUENCE
   {
   productByteSize     [1]  INTEGER               OPTIONAL,
   productFormat       [2]  InternationalString   OPTIONAL,
   productCompression  [3]  InternationalString   OPTIONAL,
   otherInfo           [4]  OtherInformation      OPTIONAL
   }

ProcessingOptions      ::=   CHOICE
   {
   formattedProcessingOptions   [1]     EXTERNAL,
   unformattedProcessingOptions [2]     InternationalString
   }

SceneSelectionOptions ::=    CHOICE
   {
   formattedSceneSelectionOptions       [1]      EXTERNAL,
   unformattedSceneSelectionOptions     [2]      InternationalString
   }
```

```
PriceInfo         ::=      SEQUENCE
   {
   price               [1]  IntUnit,
   priceExpirationDate [2]  InternationalString,
   additionalPriceInfo [3]  InternationalString   OPTIONAL
   }

OrderStatusInfo   ::=      SEQUENCE
   {
   orderState          [1]  CHOICE
                              {
                              staticState        [1] StaticState,
                              dynamicState       [2] DynamicState
                              },
   additionalStatusInfo[2]  InternationalString   OPTIONAL
   }

StaticState       ::=  [1]  IMPLICIT INTEGER
   {
   orderNotValid              (1),
   orderEstimated             (2),
   orderCompleted             (3)
   }

DynamicState      ::=  [2]  IMPLICIT INTEGER
   {
   orderBeingEstimated        (4),
   orderBeingQuoted           (5),
   orderBeingProcessed        (6),
   orderBeingCancelled        (7),
   orderBeingDeleted          (8)
   }
END
```

E.7 Collection Path Identification

The results from either a collection or product descriptor search shall include in the ***DatabaseName*** field of the *search response* or *present response* object, the full collection tree path from the original *target* collection of the search to the item descriptor located (whether it be a collection descriptor or product descriptor). Using this information a client may look for duplicate result records. The hierarchical information provided by the full collection tree path enables the client to customise the handling of duplicates according to its specific needs (i.e. the choice of one duplicate over another may be relevant for a particular client, for example, for reasons of ordering cost). This provides an extra level of service to a local catalogue.
The format of the collection tree path (i.e. as returned in a collection *search response*), shall be formatted as follows:

```
databaseName                   =    retrieval_manager_path

retrieval_manager_path         =    "/" + host + local_path

local_path                     =    "/" + collection_node + [ path ]

collection_node                =    grouping_node | ordering_node

grouping_node                  =    reg_collection_descriptor_id |
                                    unreg_collection_descriptor_id

ordering_node                  =    ordering_centre_id + "," +
                                    reg_collection_descriptor_id

reg_collection_descriptor_id   =    archive_collection_id | theme_collection_id

unreg_collection_descriptor_id =    persistent_result_set_id | persistent_query_id

path                           =    local_path | remote_path

remote_path                    =    "/" + retrieval_manager_path
```

That is, the identifier of the Retrieval Manager and its collections down to the point in the collection tree where either a terminal collection or a remote collection is encountered. If a remote collection is encountered, then there follows a double solidus ('//') plus the identifier of the remote Retrieval Manager, this itself is followed by the collection tree path in the remote Retrieval Manager, again until a terminal or remote collection is encountered. Collections within the collection path from which products can be ordered are flagged as ordering nodes with the prefix "ON_".

E.8 Task Package Names

This section specifies the definitions used in the *Extended Services facility*. The *package name* contains the *package name* proposed by the *origin*, whilst the *target* reference contains a *task package identifier* provided by the *target*. The definitions are formatted as follows:

```
packageName                    =    url_persistent_unreg_collection |
                                    url_periodic_query_id |
                                    url_database_update_id |
                                    url_CIP_order_id

persistent_unreg_collection_id =    persistent_result_set_id | persistent_query_id


url_persistent_result_set_id   =    cip_url_prefix + persistent_result_set_id

url_persistent_query_id        =    cip_url_prefix + persistent_query_id

url_periodic_query_id          =    cip_url_prefix + periodic_query_id

url_database_update_id         =    cip_url_prefix + database_update_id

url_CIP_order_id               =    cip_url_prefix + CIP_order_id


persistent_result_set_id       =    persistent_result_set_prefix + origin_timestamp

persistent_query_id            =    persistent_query_prefix + origin_timestamp

periodic_query_id              =    periodic_query_prefix + origin_timestamp

database_update_id             =    database_update_prefix + origin_timestamp

CIP_order_id                   =    CIP_order_prefix + origin_timestamp


persistent_result_set_prefix   =    "PRS_"

persistent_query_prefix        =    "QUE_"

periodic_query_prefix          =    "PQU_"

database_update_prefix         =    "DBU_"

CIP_order                      =    "ORD_"


origin_timestamp               =    the timestamp generated by the origin on
                                    requesting the package.


targetReference                =    targetHost + timestamp

target_host                    =    host

timestamp                      =    the timestamp generated by the target on
                                    creating the package.
```

E.9 Ordering Information

This section specifies the definitions used for the CIP Order facility in order to route orders to the appropriate Retrieval Manager and Local Ordering Handling System. The definitions are formatted as follows:

```
ordering_centre_id            =    target_host + LOHS_id

target_host                   =    host
                                   -- the target directly linked to the LOHS.

LOHS_id                       =    u_chars_max26
                                   -- the name of the LOHS at which the order will
                                   -- be executed.
```

E.10 Miscellaneous BNF Definitions

This section contains the low-level BNF definitions used in the format definitions in the sections above.

```
u_chars_max256                =    u_chars_max128 + [u_chars_max128]

u_chars_max128                =    u_chars_max32 + [u_chars_max32] +
                                   [u_chars_max32] + [u_chars_max32]

u_chars_max32                 =    u_chars_max8 + [u_chars_max8] +
                                   [u_chars_max8] + [u_chars_max8]

u_chars_max8                  =    u_chars_max2 + [u_chars_max2] +
                                   [u_chars_max2] + [u_chars_max2]

u_chars_max2                  =    uchar + [uchar]

digit_max5                    =    digit + [digit] + [digit] + [digit] + [digit]

u_chars_8                     =    u_char + u_char + u_char + u_char + u_char +
                                   u_char + u_char + u_char

uchar                         =    alpha ¦ digit ¦ safe ¦ extra

alpha                         =    low_alpha ¦ high_alpha

low_alpha                     =    "a" ¦ "b" ¦ "c" ¦ "d" ¦ "e" ¦ "f" ¦ "g" ¦ "h" ¦
                                   "i" ¦ "j" ¦ "k" ¦ "l" ¦ "m" ¦ "n" ¦ "o" ¦ "p" ¦
                                   "q" ¦ "r" ¦ "s" ¦ "t" ¦ "u" ¦ "v" ¦ "w" ¦ "x" ¦
                                   "y" ¦ "z"

high_alpha                    =    "A" ¦ "B" ¦ "C" ¦ "D" ¦ "E" ¦ "F" ¦ "G" ¦ "H" ¦
                                   "I" ¦ "J" ¦ "K" ¦ "L" ¦ "M" ¦ "N" ¦ "O" ¦ "P" ¦
                                   "Q" ¦ "R" ¦ "S" ¦ "T" ¦ "U" ¦ "V" ¦ "W" ¦ "X" ¦
                                   "Y" ¦ "Z"

digit                         =    "0" ¦ "2" ¦ "3" ¦ "4" ¦ "5" ¦ "6" ¦ "7" ¦ "8" ¦
                                   "9"

safe                          =    "$" ¦ "-" ¦ "_" ¦ "." ¦ "+"

extra                         =    "!" ¦ "*" ¦ "'" ¦ "(" ¦ ")" ¦ ","
```